Centre de recherche sur les transports (C.R.T.)
Centre for Research on Transportation
CRT-2004-21

VEHICLE ROUTING FOR THE HOME DELIVERY OF PERISHABLE GOODS

by

Nabila Azi
Michel Gendreau
Jean-Yves Potvin

October 2004

Université de Montréal
C.P. 6128, succursale Centre-ville, Montréal QC H3C 3J7 Canada
Téléphone : (514) 343-7575 / Télécopieur : (514) 343-7121

crt@crt.umontreal.ca
www.crt.umontreal.ca

HEC ÉCOLE POLYTECHNIQUE MONTRÉAL Université de Montréal

PUBLICATION
CRT-2004-21

VEHICLE ROUTING FOR THE HOME DELIVERY OF PERISHABLE GOODS

by

Nabila Azi[1]
Michel Gendreau[1]
Jean-Yves Potvin[1]

The research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), and by the Fonds québécois de la recherche sur la nature et les technologies (FQRNT).

1. Département d'informatique et de recherche opérationnelle and Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 (azn;michelg@crt.umontreal.ca, jean-yves.potvin@umontreal.ca)

Dépôt légal – Bibliothèque nationale du Québec
Bibliothèque nationale du Canada, 2004

105-00426/

Résumé

Dans cet article, nous décrivons un algorithme exact permettant de résoudre un problème où le même véhicule parcourt plusieurs routes afin de desservir un ensemble de clients avec fenêtres de temps. La problématique étudiée s'inspire des services de livraison à domicile pour les denrées périssables, où les routes sont courtes et doivent être enchaînées afin de créer une journée de travail pour le véhicule. Une méthode basée sur un algorithme de recherche de chemins plus courts élémentaires avec contraintes de ressources est proposée afin de résoudre le problème. La méthode se divise en deux phases : dans la première phase, toutes les routes réalisables non dominées sont générées; dans la seconde phase, certaines routes sont sélectionnées et séquencées pour former la journée de travail du véhicule. Des résultats numériques sont présentés sur les problèmes Euclidiens dérivés d'instances classiques du problème de tournées de véhicules avec fenêtres de temps.

Mots clés : tournées, fenêtres de temps, véhicules unique, routes multiples, denrées périssables, livraison à domicile, épiceries électroniques.

Abstract

This paper describes an exact algorithm for solving a problem where the same vehicle performs several routes to serve a set of customers with time windows. The motivation comes from the home delivery of perishable goods, where vehicle routes are short and must be combined to form a working day. A method based on an elementary shortest path algorithm with resource constraints is proposed to solve this problem. The method is divided into two phases: in the first phase, all non dominated feasible routes are generated; in the second phase, some routes are selected and sequenced to form the vehicle workday. Computational results are reported on Euclidean problems derived from benchmark instances of the classical vehicle routing problem with time windows.

Keywords: vehicle routing, time windows, single vehicle, multiple uses, perishable goods, home delivery, electronic groceries.

1 Introduction

In this work, we consider a variant of the Vehicle Routing Problem with Time Windows (VRPTW) where the same vehicle can perform several routes during its workday. Surprisingly, this problem has received little attention in the literature in spite of its importance in practice. For example, in the home delivery of perishable goods (e.g., food), routes are of short duration and must be combined to form a complete workday. We believe that this type of problem will become increasingly important in the future with the advent of electronic services, like e-groceries, where customers can order goods through the Internet and have them delivered at home.

The vehicle routing problem with multiple use of vehicles, but no time windows, has been addressed through heuristic means in [5, 12]. In [12], different solutions to the classical vehicle routing problem are generated using a tabu search heuristic. The routes obtained are then combined to produce working days for the vehicles by solving a bin packing problem, an idea previously introduced in [5] in the context of a savings-based heuristic. In [2], the authors introduce the home delivery problem, which is more closely related to real-world applications. Here, a probability of occurrence and a revenue are associated with each potential customer. When a new request occurs, a decision to accept or reject it must be taken in real-time, and a time window for service is determined. Although vehicle routes are generated and used to decide about the acceptance or rejection of a particular request, the "real" routes are executed later. Logistics and socio-economic considerations about different types of home delivery problems, with a particular emphasis on electronic groceries, can also be found in [6, 7, 8, 9, 13].

In this paper, a single vehicle version of a routing problem with time windows and multiple use of vehicles, which provides a natural model for many home delivery applications, is introduced. An exact problem-solving approach, based on an elementary shortest path algorithm with resource constraints, is then reported. To the best of our knowledge, this is the first exact algorithm developed for this type of problem.

The outline of the paper is as follows. In Section 2, a mathematical programming formulation is proposed. The problem-solving approach is then described in Section 3. Computational results on problem instances derived from Solomon's VRPTW testbed [11] are reported in Section 4. Finally, concluding remarks follow in Section 5.

2 Problem Formulation

The problem considered can be stated as follows. We have a single vehicle of capacity Q delivering perishable goods from a depot node to a set of customer nodes $N = \{1, 2, ..., n\}$ in a complete directed graph with arc set A. A distance d_{ij} and a travel time t_{ij} are associated with every arc $(i, j) \in A$. Each customer $i \in N$ is characterized by a demand q_i, a service or dwell time s_i and a time window $[a_i, b_i]$, where a_i is the earliest time to begin service and b_i the latest time. Accordingly, the vehicle must wait it it arrives at customer i before time a_i. The vehicle workday is made of a set of routes $K = \{1, 2, ..., k\}$ where each route starts and ends at the depot (some of these routes might be empty). We assume, without loss of generality, that the routes are served in the order 1, 2,..., k. The depot is denoted by 0 or $n + 1$ depending if it is the initial or terminal node of an arc, with $s_0 = s_{n+1} = 0$, $q_0 = q_{n+1} = 0$, $a_0 = a_{n+1} = 0$; $b_0 = b_{n+1} = \infty$; the symbol N^+ is used for $N \bigcup \{0, n+1\}$ and A^+ for $A \bigcup \{(0, n+1)\}$, where $(0, n+1)$ is a dummy arc with distance $d_{0,n+1} = 0$ and travel time $t_{0,n+1} = 0$. Due to the transportation of perishable items, every customer in a route must be served before a given deadline associated with that route. The latter is defined by

adding a constant t_{max} to the route start time. Also, a set up or vehicle loading time σ^r is associated with each route $r \in K$. The objective is to minimize the total distance traveled to serve all customers while satisfying the capacity, time window and deadline constraints.

This problem can be formulated as follows, using M as an arbitrary large constant:

$$\min \sum_{r \in K} \sum_{(i,j) \in A} d_{ij} x_{ij}^r \tag{1}$$

subject to

$$\sum_{j \in N^+} x_{ij}^r = y_i^r, \quad i \in N,\ r \in K, \tag{2}$$

$$\sum_{r \in K} y_i^r = 1, \quad i \in N, \tag{3}$$

$$\sum_{i \in N^+} x_{ih}^r - \sum_{j \in N^+} x_{hj}^r = 0, \quad h \in N,\ r \in K, \tag{4}$$

$$\sum_{i \in N^+} x_{0i}^r = 1, \quad r \in K, \tag{5}$$

$$\sum_{i \in N^+} x_{i(n+1)}^r = 1, \quad r \in K, \tag{6}$$

$$\sum_{i \in N} q_i y_i^r \leq Q, \quad r \in K, \tag{7}$$

$$t_i^r + s_i + t_{ij} - M\left(1 - x_{ij}^r\right) \leq t_j^r, \quad (i,j) \in A^+,\ r \in K, \tag{8}$$

$$a_i y_i^r \leq t_i^r \leq b_i y_i^r, \quad i \in N,\ r \in K, \tag{9}$$

$$t_0^1 \geq \sigma^1, \tag{10}$$

$$t_{n+1}^r + \sigma^{r+1} \leq t_0^{r+1}, \quad r = 1, \ldots, k-1, \tag{11}$$

$$\sigma^r = \beta \sum_{i \in N} s_i y_i^r, \quad r \in K, \tag{12}$$

$$t_i^r \leq t_0^r + t_{max}, \quad i \in N,\ r \in K, \tag{13}$$

$$x_{ij}^r \quad \textit{binary}, \quad (i,j) \in A^+,\ r \in K, \tag{14}$$

$$y_i^r \quad \textit{binary}, \quad i \in N,\ r \in K, \tag{15}$$

where

- x_{ij}^r is 1 if arc $(i,j) \in A^+$ is in route r, 0 otherwise; note that $x_{0,n+1}^r$ is 1 if route r is empty ;
- y_i^r is 1 if customer i is in route r, 0 otherwise;
- t_i^r is the time of beginning of service at customer i in route r;
- t_0^r is the start time of route r;
- t_{n+1}^r is the end time of route r.

In this formulation, equation (3) states that every customer should be visited exactly once. Equations (4), (5) and (6) are flow conservation constraints that describe the vehicle path. Equation (7) states that the total demand on a route should not exceed vehicle capacity. Equations (8), (9), (10) and (11) ensure feasibility of the time schedule. Equation (12) defines the vehicle loading time as the sum of service times of all customers in a route, multiplied by parameter β. Finally, equation (13) corresponds to the deadline constraint for serving a customer. Note that equation (9) forces the t_i^r variables to 0 when customer i is not in route r. Consequently, equation (13) is automatically satisfied in this case.

In practice, it might not be possible to serve all customers with the vehicle due to the time window constraints. An alternative objective is thus to maximize the number of served customers and, for the same number of served customers, to minimize the total distance. The objective in the above formulation can be modified to account for the number of served customers, namely:

$$\min \sum_{r \in K} \sum_{(i,j) \in A} d_{ij} x_{ij}^r - \alpha \sum_{r \in K} \sum_{i \in N} y_i^r. \tag{16}$$

By setting the weighting parameter α to a sufficiently large value, the desired hierarchic objective is obtained. Note also that the equality sign in constraint (3) should be replaced by a $\leq$ sign. The latter problem is the one that is considered in the following.

3 Problem-Solving Approach

The problem is addressed via an approach that exploits an elementary shortest path algorithm with resource constraints [4], noted FDGG in the following. This algorithm is briefly introduced in subsection 3.1. Then, our problem-solving methodology is described.

3.1 An elementary shortest path algorithm with resource constraints

FDGG is a label correcting algorithm that solves the elementary shortest path problem with resource constraints on graphs with negative cycles. In this context, a path is characterized by the consumption of each resource, in addition to its length. Accordingly, when different paths lead to the same node, one path dominates the others only if it is shorter and consumes less resources, for every resource considered. As a consequence, many different labels are typically maintained at each node (i.e., all non dominated paths leading to that node).

In the case of FDGG, negative cycles must also be avoided. Thus, a path p from some origin node o to some node j is labeled with $R_p = (d_p, t_p^1, \ldots, t_p^l, s_p, V_p^1, \ldots, V_p^n)$, where

$L = \{1, ..., l\}$ is the set of resources, d_p is the length of path p, t_p^k is the consumption of resource $k = 1, ..., l$, s_p is the number of unreachable nodes (either because they have already been visited or because their inclusion would violate one or more resource constraints) and $V_p^i = 1$ if node i is unreachable, 0 otherwise. The following dominance criterion is then defined:

Dominance criterion. If p and p' are two different paths from origin o to node j with labels R_p and R'_p, respectively, then path p dominates p' if and only if $d_p \leq d_{p'}$, $s_p \leq s_{p'}$, $t_p^k \leq t_{p'}^k$, $k = 1, ..., l$, $V_p^i \leq V_{p'}^i$, $i = 1, ..., n$ and $R_p \neq R'_p$.

That is, path p dominates p' if (1) it is shorter, (2) it consumes less resources for every resource considered and (3) every unreachable node is also unreachable for path p'. Note that there might be an equality between the two paths for some of these, but at least one inequality must strictly hold. As stated in [4], by eliminating paths through this dominance relation, only labels corresponding to non dominated elementary paths are kept and a solution to the problem is obtained at the end.

Our problem-solving approach, based on this algorithm, is divided into two phases. In the first phase, feasible routes are generated. Then, some of these routes are combined to form a workday for the vehicle. These two phases are described in the following.

3.2 Phase 1

First, we want to generate a set of feasible routes that will be used as "building blocks" in the second phase. As the deadline constraint is quite restrictive in practice, even the largest feasible routes will contain only a few customers and a pure enumerative approach is viable. Through a suitable adaptation of the FDGG algorithm, however, it is possible to discard partial routes that are of no interest due to the availability of better alternative routes.

First, the distance d_{ij} on each arc is replaced by $d_{ij} - \alpha$, with $\alpha > max_{(i,j)\in A}\ d_{ij}$. As it is always beneficial to extend a path with a new arc, the corresponding distance being negative, this modification allows the FDGG algorithm to generate closed paths that start and end at the depot (otherwise, it would be optimal to stay at the depot). Second, as our goal is not to find shortest routes, but rather to generate feasible routes, condition (3) in the FDGG dominance criterion is modified and now states that the nodes visited in one path should be the same as those visited in the other path. Using the notation $W_p^i = 1$ if node i is in path p, 0 otherwise, the dominance criterion is now defined as follows:

Dominance criterion (modified). If p and p' are two different paths from origin o to node j with labels R_p and R'_p, respectively, then path p dominates p' if and only if $d_p \leq d_{p'}$, $t_p^k \leq t_{p'}^k$, $k = 1, ..., l$, $W_p^i = W_{p'}^i$, $i = 1, ..., n$ and $R_p \neq R'_p$.

A partial path is thus discarded when another leads to the same node, going through the same intermediary nodes (although in a different order) and is of smaller length and consumes less resources. In this application, the resource constraints are the time windows and route deadlines. Clearly, there is no interest in extending dominated paths, because we always want to visit a given subset of customers in the most efficient way. All feasible non dominated closed routes obtained with this modified FDGG algorithm are kept and used in Phase 2 to generate a solution to our problem.

3.3 Phase 2

The FDGG algorithm is used again in Phase 2 to create a workday for the vehicle. Here, the original algorithm is applied on a transformed graph where the nodes correspond to the

routes generated in Phase 1, plus two artificial nodes that correspond to the start and end of the vehicle workday. In the graph, there is an arc between nodes r and r' if (1) the two subsets of customers in routes r and r' are disjoint and (2) it is feasible to serve route r' after route r, where feasibility is determined through departure time windows (see below). The cost on arc (r, r') relates to the objective in equation (16) for route r' and is set to the length of route r' minus the weighting parameter α times the number of customers in route r'. That is,

$$\sum_{(i,j)\in A} d_{ij} x_{ij}^{r'} - \alpha \sum_{i\in N} y_i^{r'}, \tag{17}$$

with $\alpha > 2max_{i\in N}\{max\{d_{0i}, d_{i,n+1}\}\}$. With this requirement on parameter α, and assuming that the triangular inequality holds, the costs are all negative. Namely,

$$\begin{aligned}
\sum_{(i,j)\in A} d_{ij} x_{ij}^{r'} &\leq \sum_{i\in N} (d_{0i} + d_{i,n+1}) y_i^{r'} \qquad \text{(triangular inequality)} \\
&\leq \sum_{i\in N} 2max\{d_{oi}, d_{i,n+1}\} y_i^{r'} \\
&\leq 2\ max_{i\in N}\{max\{d_{oi}, d_{i,n+1}\}\} \sum_{i\in N} y_i^{r'} \\
&< \alpha \sum_{i\in N} y_i^{r'}.
\end{aligned}$$

Thus, it is always beneficial to serve an additional customer if it is feasible to do so. In the transformed graph, there is also an arc from the artificial start node to every route node and from every route node to the artificial end node (in the latter case, the costs are set to 0).

As previously mentioned, time windows are associated with route nodes. These are calculated as follows.

3.3.1 Latest departure and arrival times

Let us assume that route r corresponds to the sequence $(0 = i_0, i_1, i_2, ..., i_{n_r}, i_{n_r+1} = n+1)$, where n_r is the number of customers in the route. We first need to determine the latest feasible time $\overline{t}_{i_j}^r$ to begin service at each customer (where $\overline{t}_{i_0}^r = \overline{t}_0^r$ and $\overline{t}_{i_{n_r+1}}^r = \overline{t}_{n+1}^r$ are the latest feasible departure and arrival times at the depot, respectively). A backward sweep of route r is applied from i_{n_r+1} to i_0 as follows:

$$\overline{t}_{i_{n_r+1}}^r \leftarrow b_{i_{n_r+1}},$$

$$\overline{t}_{i_j}^r \leftarrow min\{\overline{t}_{i_{j+1}}^r - t_{i_j i_{j+1}} - s_{i_j},\ b_{i_j}\},\ j = i_{n_r}, ..., i_0 = 0.$$

Once $\overline{t}_{i_0}^r$ has been obtained, a forward sweep is applied to reset the $\overline{t}_{i_j}^r$ values and get the latest feasible schedule:

$$\overline{t}_{i_j}^r \leftarrow max\{\overline{t}_{i_{j-1}}^r + s_{i_{j-1}} + t_{i_{j-1} i_j},\ a_{i_j}\},\ j = i_1, ..., i_{n_r+1} = n+1.$$

The total route duration corresponds to $d^r_{min} = \overline{t}^r_{n+1} - \overline{t}^r_0$, which is also the minimum duration, because the waiting time is minimized by serving the route at the latest feasible time.

3.3.2 Earliest departure and arrival times

We now want to determine the earliest departure and arrival times at the depot, $\underline{t}^r_{i_0} = \underline{t}^r_0$ and $\underline{t}^r_{i_{n_r+1}} = \underline{t}^r_{n+1}$, while maintaining the minimum route duration d^r_{min}. Two cases should be considered:

Case 1. If there is no waiting time in the latest feasible schedule, then the latter can be "shifted" by δ^r time units, where:

$$\delta^r = min_{j=0,\ldots,n_r+1}\ \overline{t}^r_{i_j} - a_{i_j}.$$

The time windows for arrival and departure at the depot are thus $[\underline{t}^r_0, \overline{t}^r_0]$ and $[\underline{t}^r_{n+1}, \overline{t}^r_{n+1}]$, where $\underline{t}^r_0 = \overline{t}^r_0 - \delta^r$ and $\underline{t}^r_{n+1} = \overline{t}^r_{n+1} - \delta^r$.

Case 2. If there is some waiting time in the latest feasible schedule, it is not possible to depart earlier from the depot without increasing route duration. Consequently, we have $\underline{t}^r_0 = \overline{t}^r_0$ and $\underline{t}^r_{n+1} = \overline{t}^r_{n+1}$. In this case, the time windows for departure and arrival at the depot reduce to a single point. It is thus feasible to serve route r' after route r, and arc (r, r') is in the route graph, if $\underline{t}^r_{n+1} + \sigma^{r'} \leq \overline{t}^{r'}_0$ (assuming that the two routes are disjoint).

The original FDGG algorithm is used to find an elementary least cost path with time window constraints from the artificial start node to the artificial end node in the route graph. The departure time windows associated with each route are used for this purpose. That is, the vehicle must be back at the depot and ready to depart before the latest departure time of its next route. If the vehicle is ready before the earliest departure time, then it must wait. When arc (r, r') is added to the current path, route r' is included into the vehicle workday. In this case, the cost of route r' in equation (17) is incurred and the time consumed corresponds to the duration of route r', plus the loading time of route r', plus any waiting time before departure. The sequence of nodes in the least cost path obtained at the end corresponds to the sequence of routes in the vehicle workday. An example is provided below.

3.4 An example

Let us assume that we have the depot 0 and 5 customers indexed from 1 to 5 in the Euclidean plane and that distances and travel times are the same. Table 1 indicates their coordinates and time windows (taken from Solomon's test instances [11]).

Now, we want to calculate the departure and arrival time windows for route $r = 1$, $0 - 3 - 1 - 6$, where $i_0 = 0$, $i_1 = 3$, $i_2 = 1$ and $i_3 = 6$ and where nodes 0 and 6 stand for the depot. With travel times $t_{03} = 39.35$, $t_{31} = 3.00$ and $t_{16} = 38.07$ and a service time of 10 at customers 3 and 1, we have:

$\overline{t}^1_6 = \infty$,
$\overline{t}^1_1 = min\{\infty - 38.07 - 10, 874\} = 874$,
$\overline{t}^1_3 = min\{874 - 3.00 - 10, 588\} = 588$,
$\overline{t}^1_0 = min\{588 - 39.35, \infty\} = 548.65$.

Now,

Node	x	y	time window
0	40	50	[0, ∞]
1	25	85	[591, 874]
2	22	75	[73, 350]
3	22	85	[473, 588]
4	20	80	[418, 913]
5	20	85	[40, 390]

Table 1: Customers with their time windows

$\bar{t}_0^1 = 548.65,$
$\bar{t}_3^1 = max\{548.65 + 39.35, 473\} = 588,$
$\bar{t}_1^1 = max\{588 + 3.00 + 10, 591\} = 601,$
$\bar{t}_6^1 = max\{601 + 38.07 + 10, 0\} = 649.07.$

Thus, the minimum route duration is equal to 649.07 - 548.65 = 100.42. Since there is no waiting time in this schedule, we have

$$\delta^1 = min\{548.65 - 0, 588 - 473, 601 - 591, 649.07 - 0\} = 10.$$

Consequently, the time windows for departure and arrival at the depot are $[\underline{t}_0^1, \bar{t}_0^1] = [538.65, 548.65]$ and $[\underline{t}_6^1, \bar{t}_6^1] = [639.07, 649.07]$, respectively. In Table 2, we show the time windows for all feasible routes generated in Phase 1, plus the costs obtained by setting α to $2\ max_{i \in N}\{max\{d_{0i}, d_{i,n+1}\}\} + 1$. With these, we can construct the route graph illustrated in Figure 1 (where only one arc going out of the artificial start node and only one arc going into the artificial end node are shown, namely those that are part of the shortest path). The elementary shortest path obtained with FDGG is shown with a dotted line. The workday of the vehicle is thus made of route $0-2-5-6$ followed by routes $0-4-6$ and $0-3-1-6$.

Route	Departure	Arrival	Cost
0 3 1 6	[538.65 , 548.65]	[639.07, 649.07]	-82.82
0 4 3 6	[421.57 , 522.35]	[536.57, 637.35]	-82.46
0 4 1 6	[537.88 , 639.07]	[820.88, 922.07]	-82.05
0 2 5 6	[42.20 , 143.50]	[319.20, 420.50]	-81.94
0 2 6	[42.20 , 113.80]	[319.20, 390.80]	-20.02
0 4 6	[381.95 , 464.05]	[876.95, 959.05]	-9.52
0 1 6	[552.93 , 639.07]	[35.93, 922.07]	-5.48
0 3 6	[433.65 , 522.35]	[548.65, 637.35]	-2.92
0 5 6	[0.00 , 90.62]	[349.69, 440.31]	-1.00

Table 2: Time windows and cost of each route

4 Computational Results

We have used the classical VRPTW instances of Solomon [11] for this study. Those are 100-customer Euclidean problems where distances and travel times are the same. There are six different classes of problems depending on the geographic location of customers (R:

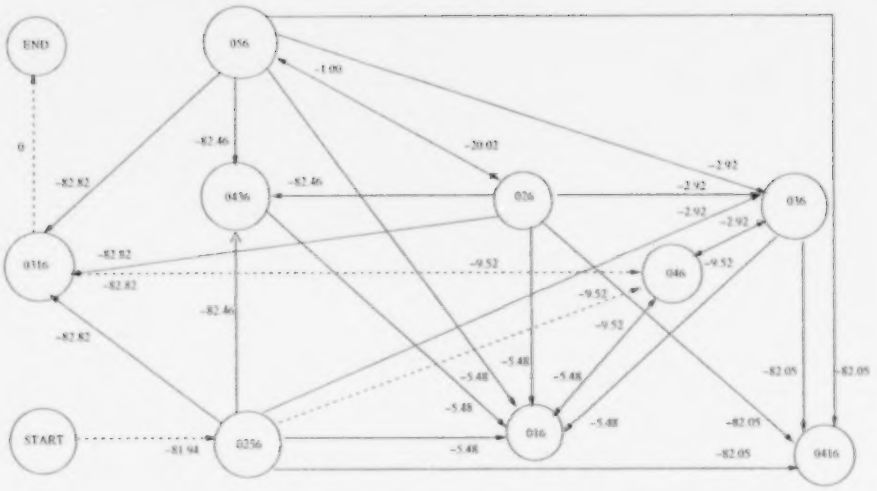

Figure 1: Route graph

random; C: clustered; RC: mixed) and length of scheduling horizon (1: short horizon; 2: long horizon). In this study, problems of type 1 have been discarded because the short horizon does not allow the vehicle to serve many routes during its workday. We thus report results for problem classes R2 (11 instances), C2 (8 instances) and RC2 (8 instances) only. All tests were run on a 900 MHz Sun Ultra III with 8 Gig of RAM.

Deadline constraints were added to Solomon's instances. Two different t_{max} values were tested for each problem class to evaluate how this parameter impact the algorithmic behavior. In the case of R2 and RC2, t_{max} was set to 75 and 90, while in the case of C2, it was set to 220 and 240. These values had to be carefully chosen: if too small, the feasible routes would contain one or two customers at most; if too large, the problem would not be tractable due to an excessive number of feasible routes. The t_{max} value also needs to be larger for class C2 because the service time at each customer is 30, as opposed to 10 for classes R2 and RC2. It should finally be noted that parameter β for the route loading time in equation (12) was set to 0.2 in all experiments.

4.1 Impact of dominance criterion in Phase 1

In this section, we first evaluate the impact of the modified dominance criterion when feasible routes are generated with the elementary shortest path algorithm with resource constraints. Basically, we compare the number of routes produced with and without the dominance criterion. In the latter case, an exhaustive enumeration of all feasible routes is obtained, as no dominance relation exists. Table 3 shows the average number of routes obtained on classes R2, RC2 and C2, using t_{max}=90, 90 and 240, respectively.

As we can see, by exploiting the dominance criterion in the shortest path algorithm, the number of feasible routes generated in Phase 1 is reduced by approximately 20% (the routes that are eliminated are feasible, but are of no interest, as they are dominated by other routes). Clearly, this can only marginally impact the "combinatorial explosion" associated with increasing t_{max} values, but this is a nice way to exploit the shortest path algorithm

already available to us. Note also that the number of feasible routes is much smaller for classes C2 and RC2, when compared with class R2. This is related to the presence of distant clusters in C2 and RC2 that prevent feasible routes to serve customers in different clusters.

Problem class	Without criterion	With criterion	Reduction in percent
R2	30695.8	21984.1	28.4%
C2	2145.6	1732.3	19.3%
RC2	4384.3	3450.5	21.3%

Table 3: Number of routes generated in Phase 1.

4.2 Results on Solomon's instances

In Tables 4 to 9, the results obtained on Solomon's instances are reported. These results include smaller instances obtained by taking only the first 25 or 50 customers. In these tables, a particular instance is identified by its class, followed by its number (1 to 8, or 1 to 11, depending on the class) and number of customers. For example RC202.50 is the second instance of class RC2, where only the first 50 customers are considered. The other columns contain the number of feasible routes generated in Phase 1 (#Routes Phase 1), number of routes in the vehicle workday at the end of Phase 2 (#Routes Phase 2), total distance traveled by the vehicle, number of served customers, percentage of served customers, average number of customers per route and CPU time in seconds. When an entry is empty, the corresponding instance could not be solved. In some cases, we were able to generate feasible routes in Phase 1, but the associated route graph was too large to allow us to produce a solution.

Clearly, the instances in class R2 are harder to solve, due to a large number of feasible routes. In fact, only one solution was obtained on a 100-customer instance (R201). In the case of classes C2 and RC2, solutions were obtained on all instances with t_{max}=75 and t_{max}=220, 240, respectively. By increasing the value of t_{max}, solutions with fewer routes, more customers per route and a larger number of served customers are obtained. However, the number of feasible routes increases sharply, thus leading to a substantial increase in computational requirements. In the case of class RC2, for example, results were obtained on all problem instances with $t_{max} = 75$, but a single 100-customer instance was solved with $t_{max} = 90$. Overall, the CPU times vary widely and range from only a few seconds to several days, depending on the problem size, problem class and t_{max} value.

It should be noted that, on the 100-customer instances, approximately 30% of the customers are served. This percentage increases to about 50%-60% on the 50-customer instances and to 70%-100% on the smallest instances with 25 customers. The small percentages observed in the case of the 100-customer instances is not a surprise, given that the latter have been designed for multiple vehicles. That is, when we consider a relatively short time period within the scheduling horizon and gather all customers with a time window that falls into that period, there are clearly too many for a single vehicle to serve them all (while this could be possible with many vehicles).

Instance	# Routes Phase 1	# Routes Phase 2	Distance	# Cust. served	% Cust. served	# Cust. per route	CPU time (sec)
R201.25	122	8	315.3	19	76%	2.37	1
R201.50	550	9	440.2	25	50%	2.77	20
R201.100	4279	11	379.8	36	36%	3.27	16156
R202.25	341	9	337.6	20	80%	2.22	3
R202.50	2223	10	430.6	28	56%	2.80	1075
R202.100	32495						
R203.25	443	7	376.4	20	80%	2.85	18
R203.50	3248	9	434.9	30	60%	3.33	8081
R203.100	62685						
R204.25	547	8	407.8	23	92%	2.87	45
R204.50	4535	9	448.5	32	64%	3.55	33068
R204.100	87542						
R205.25	237	9	434.0	23	92%	2.55	4
R205.50	1424	9	457.1	29	57%	3.22	596
R205.100	16399						
R206.25	425	9	416.2	22	88%	2.44	11
R206.50	3059	10	463.3	30	60%	3.00	4078
R206.100	50575						
R207.25	480	9	441.9	24	96%	2.66	20
R207.50	3682	10	422.0	32	64%	3.20	11326
R207.100	73008						
R208.25	552	8	420.0	24	96%	3.00	44
R208.50	4548	9	430.8	32	64%	3.55	34041
R208.100	93918						
R209.25	381	9	466.0	23	92%	2.55	11
R209.50	2531	8	421.9	28	56%	3.50	2754
R209.100	42731						
R210.25	397	8	429.4	21	84%	2.62	11
R210.50	3160	9	440.3	29	57%	3.22	5051
R210.100	51537						
R211.25	597	9	412.7	24	96%	2.66	41
R211.50	5067	9	460.6	30	60%	3.33	19588
R211.100	124008						

Table 4: Results on class R with $t_{max} = 75$.

Instance	# Routes Phase 1	# Routes Phase 2	Distance	# Cust. served	% Cust. served	# Cust. per route	CPU time (sec)
R201.25	217	8	392.5	20	80%	2.50	1
R201.50	1263	9	419.7	26	52%	2.88	203
R201.100	13729	10	404.4	37	37%	3.7	245906
R202.25	778	7	304.5	20	80%	2.85	26
R202.50	6800	8	452.8	29	57%	3.62	16410
R202.100							
R203.25	1097	6	397.1	21	84%	3.50	98
R203.50	11320	9	443.1	33	66%	3.66	141630
R203.100							
R204.25	1435	6	361.3	22	88%	3.66	385
R204.50	7209	8	476.1	33	66%	4.1	82361
R204.100							
R205.25	527	7	351.6	22	88%	3.14	21
R205.50	4003	8	450.9	29	57%	3.62	8441
R205.100							
R206.25	1034	8	459.6	24	96%	3.00	89
R206.50	10491	8	482.2	32	64%	4.00	100630
R207.100							
R207.25	1236	7	413.0	23	92%	3.28	221
R207.50	13539	8	417.6	33	66%	4.12	316096
R207.100							
R208.25	1462	6	384.6	23	92%	3.83	534
R208.50	17930						
R208.100							
R209.25	902	8	425.6	23	92%	2.87	64
R209.50	8245	9	441.5	30	60%	3.33	46973
R209.100							
R210.25	986	8	441.4	22	88%	2.75	67
R210.50	11144	8	450.4	30	60%	3.75	113942
R210.100							
R211.25	1649	8	426.1	25	100%	3.12	422
R211.50	8123	9	461.7	31	62%	3.4	60820
R211.100							

Table 5: Results on class R with $t_{max} = 90$.

Instance	# Routes Phase 1	# Routes Phase 2	Distance	# Cust. served	% Cust. served	# Cust. per route	CPU time (sec)
RC201.25	74	8	339.1	16	64%	2.00	1
RC201.50	174	7	340.2	19	38%	2.71	1
RC201.100	1562	10	426.2	28	28%	2.80	877
RC202.25	167	7	328.9	18	72%	2.57	1
RC202.50	395	7	358.0	22	44%	3.14	10
RC202.100	5210	10	439.3	32	32%	3.20	24561
RC203.25	252	7	327.5	20	80%	2.85	3
RC203.50	587	7	342.1	24	48%	3.42	47
RC203.100	9877	8	398.7	33	33%	4.12	188029
RC204.25	320	8	407.3	21	84%	2.62	7
RC204.50	866	7	325.3	24	48%	3.42	160
RC204.100	9141	9	396.1	34	34%	3.77	202461
RC205.25	147	7	348.2	18	72%	2.57	1
RC205.50	353	7	353.8	22	44%	3.14	6
RC205.100	4036	9	383.3	30	30%	3.33	9389
RC206.25	138	7	327.4	18	72%	2.57	1
RC206.50	334	8	381.5	23	46%	2.87	6
RC206.100	4966	9	366.4	31	31%	3.44	23338
RC207.25	258	8	385.4	20	80%	2.50	2
RC207.50	680	7	366.8	24	48%	3.42	40
RC207.100	11016	9	420.9	32	32%	3.55	278804
RC208.25	386	8	370.7	22	88%	2.75	10
RC208.50	998	8	364.4	25	50%	3.12	196
RC208.100	10640	9	379.4	32	32%	3.55	188745

Table 6: Results on class RC with $t_{max} = 75$.

Instance	# Routes Phase 1	# Routes Phase 2	Distance	# Cust. served	% Cust. served	# Cust. per route	CPU time (sec)
RC201.25	132	8	342.9	17	68%	2.12	0
RC201.50	312	6	329.0	21	42%	3.50	5
RC201.100	4456	10	393.9	29	28%	2.90	15449
RC202.25	323	8	356.3	20	80%	2.50	4
RC202.50	911	7	378.5	26	52%	3.71	74
RC202.100	20000						
RC203.25	493	7	332.0	22	88%	3.14	13
RC203.50	1413	7	347.2	28	56%	4.00	723
RC203.100	45811						
RC204.25	708	7	350.0	23	92%	3.28	64
RC204.50	2404	7	348.1	30	60%	4.28	5175
RC204.100	89808						
RC205.25	264	7	359.8	20	80%	2.85	2
RC205.50	753	7	364.7	24	48%	3.42	54
RC205.100	15161						
RC206.25	259	6	324.7	20	80%	3.33	5
RC206.50	727	7	370.1	25	50%	3.57	73
RC206.100	18989						
RC207.25	560	7	349.5	21	84%	3.00	18
RC207.50	1792	7	359.3	25	50%	3.57	844
RC207.100	53233						
RC208.25	954	7	384.9	23	92%	3.28	136
RC208.50	3117	7	364.0	27	54%	3.85	4800
RC208.100	115098						

Table 7: Results on class RC with $t_{max} = 90$.

Instance	# Routes Phase 1	# Routes Phase 2	Distance	# Cust. served	% Cust. served	# Cust. per route	CPU time (sec)
C201.25	102	10	326.4	22	88%	2.20	1
C201.50	340	12	356.6	28	56%	2.33	10
C201.100	1204	11	320.9	30	30%	2.72	547
C202.25	287	12	374.6	24	96%	2.00	6
C202.50	1024	12	384.1	29	57%	2.41	366
C202.100	3558	11	311.0	30	30%	2.72	18420
C203.25	410	12	391.0	24	96%	2.00	19
C203.50	1669	11	379.6	29	57%	2.63	1820
C203.100	5701	11	343.2	30	30%	2.72	84107
C204.25	525	10	382.4	25	100%	2.50	33
C204.50	2129	11	356.2	30	60%	2.72	4330
C204.100	7319	11	325.0	31	31%	2.81	225525
C205.25	140	10	341.0	24	96%	2.40	1
C205.50	548	10	322.2	28	56%	2.80	45
C205.100	2047	10	290.1	30	30%	3.00	2137
C206.25	166	11	367.2	25	100%	2.27	1
C206.50	718	11	349.8	29	57%	2.63	99
C206.100	2641	11	289.8	31	31%	2.81	5056
C207.25	282	10	377.6	24	96%	2.40	11
C207.50	1066	12	370.7	30	60%	2.50	549
C207.100	3078	11	313.9	31	31%	2.81	13401
C208.25	210	10	382.4	25	100%	2.50	5
C208.50	882	12	346.2	30	60%	2.50	299
C208.100	3244	11	292.8	31	31%	2.81	15586

Table 8: Results on class C with $t_{max} = 220$.

Instance	# Routes Phase 1	# Routes Phase 2	Distance	# Cust. served	% Cust. served	# Cust. per route	CPU time (sec)
C201.25	139	9	293.8	23	92%	2.55	1
C201.50	505	10	320.7	28	56%	2.80	46
C201.100	1804	10	316.6	30	30%	3.00	1906
C202.25	634	9	303.9	24	96%	2.66	39
C202.50	2461	10	353.0	29	57%	2.90	3707
C202.100	9064	11	326.5	31	31%	2.81	213601
C203.25	1288	9	298.1	24	96%	2.66	251
C203.50	5171	11	386.7	30	60%	2.72	28996
C203.100							
C204.25	1661	9	316.7	25	100%	2.77	562
C204.50	7405	10	333.1	30	60%	3	109046
C204.100							
C205.25	269	9	314.4	25	100%	2.77	3
C205.50	965	10	318.5	29	57%	2.90	195
C205.100	3575	10	286.8	30	30%	3.00	9275
C206.25	334	9	309.5	25	100%	2.77	8
C206.50	1428	10	358.9	30	60%	3.00	782
C206.100	5182	11	283.7	31	31%	2.81	34179
C207.25	639	10	340.4	25	100%	2.50	78
C207.50	2518	11	343.3	30	60%	2.72	6107
C207.100	6420	11	297.0	31	31%	2.81	97259
C208.25	416	9	309.5	25	100%	2.77	20
C208.50	1847	10	325.8	30	60%	3.00	1996
C208.100	6881	11	283.9	31	31%	2.81	110911

Table 9: Results on class C with $t_{max} = 240$.

4.3 Results on individual routes

In this subsection, we consider instances created from individual routes in the best known solutions for class RC2 (as reported in [1, 3, 10]). That is, our algorithm was applied only to the subset of customers associated with a particular route. As a feasible route in a VRPTW solution is made of customers with time windows that are well distributed over the time horizon, we expect our algorithm to include a large fraction of these customers in the vehicle workday. Overall, the new created instances contain between 21 and 44 customers.

Results obtained with $t_{max} = 90$ are shown in Table 10. In this table, an identifier like RC201.21 is the instance created with the 21 customers taken from one of the routes in the best know solution for RC201. As we can see, the number of customers per route has substantially increased, when compared with the 25-customer instances of subsection 4.2. Also, the percentages of served customers now range between 85% and 100% on the instances with 30 customers or less, which is a clear improvement over the corresponding 25-customer instances of subsection 4.2. In fact, we could not expect much better, given the additional deadline constraint that forces the vehicle to travel back to the depot and reload, thus leading to a waste of time.

Instance	# Routes Phase 1	# Routes Phase 2	Distance	# Cust. served	% Cust. served	# Cust. per route	CPU time (sec)
RC201.21	1456	4	335.7	20	95%	5.00	80
RC201.27	2690	5	375.7	24	88%	4.80	743
RC201.26	1114	5	367.6	23	88%	4.60	101
RC201.26	4087	5	321.6	23	88%	4.60	1301
RC202.38	8970	7	441.6	28	73%	4.00	26364
RC202.30	2546	7	427.4	26	86%	3.71	786
RC202.32	5922	7	389.0	26	81%	3.71	6950
RC203.27	6013	5	313.7	23	85%	4.60	4470
RC203.29	4618	7	453.0	25	86%	3.57	5689
RC203.44	6279	7	423.6	29	65%	4.14	35398
RC204.34	8768	7	386.6	28	82%	4.00	36034
RC204.32	9352	8	471.5	30	93%	3.75	119237
RC204.34	9407	7	446.2	30	88%	4.28	76790
RC205.29	5609	6	358.8	25	86%	4.16	7588
RC205.22	1338	5	370.6	20	90%	4.00	86
RC205.28	3687	6	443.2	24	85%	4.00	938
RC205.21	4702	5	307.2	21	100%	4.20	1450
RC206.32	17700	6	453.4	28	87%	4.66	71160
RC206.35	11487	6	426.4	27	77%	4.50	88764
RC206.33	11908	6	348.6	27	81%	4.50	49263
RC207.27	8145	6	371.0	26	96%	4.33	30034
RC207.35	11492	6	405.5	28	80%	4.66	86864
RC207.38	5759	6	416.3	26	68%	4.33	13630
RC208.34	6816	7	407.1	25	73%	3.57	33726
RC208.33	5804	6	360.7	29	87%	4.83	27404
RC208.33	6350	6	436.7	26	78%	4.33	25298

Table 10: Results on class RC for individual routes.

5 Conclusion

This paper has described an exact algorithm for solving a routing problem where a vehicle performs several routes over the scheduling horizon. The results indicate that this algorithm

is very sensitive to the deadline constraint, which is not a surprise. When this constraint is not tight enough, the number of feasible routes "explodes" and becomes too large to allow the algorithm to produce a solution. Future developments will now be aimed at considering problems that are closer to real-world applications. First, a fleet of many vehicles will be considered. Then, a dynamic version of the problem will be tackled, where new incoming customer requests have to be integrated in real time into the current solution.

Acknowledgments. Financial support for this work was provided by the Canadian Natural Sciences and Engineering Research Council (NSERC) and by the Quebec Fonds pour la Formation de Chercheurs et l'Aide à la Recherche (FCAR). This support is gratefully acknowledged.

References

[1] Bent R., Van Hentenryck P., "A Two-Stage Hybrid Local Search for The Vehicule Routing Problem with Time Windows", Technical Report CS-01-06, Computer Science Department, Brown University, 2001.

[2] Campbell A.M., Savelsbergh M., "Decision Support for Consumer Direct Grocery Initiatives", Report TLI-02-09, Georgia Institute of Technology, 2002.

[3] Czech Z.J., Czarnas, P., "A Parallel Simulated Annealing for the Vehicule Routing Problem with Time Windows", in *Proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing*, Canary Islands, Spain, 376–383, 2002.

[4] Feillet D., Dejax P., Gendreau M., Gueguen C., "An Exact Algorithm for the Elementary Shortest Path Problem with Resource Constraints: Application to Some Vehicle Routing Problems", Technical Report CRT-2000-15, Centre de recherche sur les transports, Montréal, 2000.

[5] Fleischmann B., "The Vehicle Routing Problem with Multiple Use of Vehicles", Working Paper, Fachbereich Wirtschaftswissenschaften, Universität Hamburg, 1990.

[6] Kilpala H.K., "The Impact of Electronic Commerce on Transport & Logistics in the Retail Grocery Industry", M.Sc. Thesis, University of Oulu, Finland, 1999.

[7] Lin I.L., Mahmassani H.S., "Can online grocers deliver? Some Logistics Considerations.", *Transportation Research Record 1817*, 17–24, 2002.

[8] Punaviki M., Saranen J., "Identifying the Success Factors in E-Grocery Home Delivery", *International Journal of Retail and Distribution Management 29*, 156–163, 2001.

[9] Punaviki M., "Comparing Alternative Home Delivery Models for E-Grocery Business", Doctoral Dissertation, Department of Industrial Engineering and Management, Helsinki University of Technology, Finland, 2003.

[10] Rousseau L.-M., Gendreau M., Pesant G., "Using Constraint-Based Operators to Solve the Vehicle Routing Problem with Time Windows", Technical Report CRT-2003-10, Centre de recherche sur les transports, Montreal, 2003.

[11] Solomon M.M., "Algorithms for the Vehicle Routing and Scheduling Problem with Time Window Constraints", *Operations Research 35*, 254–265, 1987.

[12] Taillard É. D., G. Laporte, M. Gendreau, "Vehicle Routeing with Multiple Use of Vehicles", *Journal of the Operational Research Society 47*, 1065-1070, 1996.

[13] Yrjölä H., "Physical Distribution Considerations for Electronic Grocery Shopping", *International Journal of Physical Distribution and Logistics Management 31*, 746–761, 2001.